

(19)



JAPANESE PATENT OFFICE

PATENT ABSTRACTS OF JAPAN

(11) Publication number: **2001069104 A**(43) Date of publication of application: **16.03.01**

(51) Int. Cl.

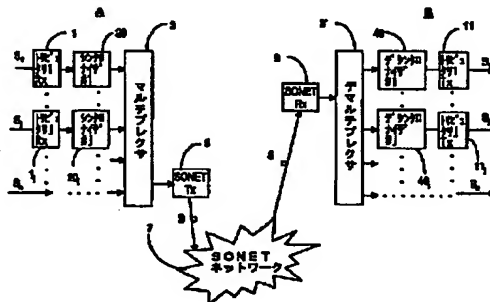
H04J 3/00(21) Application number: **2000204447**(22) Date of filing: **06.07.00**(30) Priority: **08.07.99 US 99 349087**(71) Applicant: **NORTEL NETWORKS LTD**(72) Inventor: **ROBERTS KIM B**(54) **METHOD FOR MAPPING ARBITRARY SIGNAL IN SONET**

COPYRIGHT: (C)2001,JPO

(57) Abstract:

PROBLEM TO BE SOLVED: To map arbitrary successive signals in a SONET frame by providing stages or the like for dispersing bits of the successive signals to effective time slots of frames of the container signal at a transmission site and imparting the stuff bits to the ineffective time slots which are uniformly dispersed to the frames.

SOLUTION: After signals S1..., Sj..., Sn... are mapped in respective SONET envelopes, tributaries are multiplexed by a SONET multiplexer 3 into a high-rate signal S, which is sent to a site B by a transmitter 5 through an optical network 7. Then the fixed-length container signal having a rate higher than the arbitrary rate of the successive signals is selected, the bits of the successive signals are dispersed to effective time slots of frames of the container signal at a transmission site A, and the stuff bits are given to the ineffective time slots which are uniformly dispersed to the frames.



1

(19) 日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開 2001-69104

(P 2001-69104 A)

(43) 公開日 平成13年3月16日 (2001. 3. 16)

(51) Int. Cl. ⁷

H 0 4 J 3/00

識別記号

F I

H 0 4 J 3/00

テーマコード (参考)

U

審査請求 未請求 請求項の数 20

OL 外国語出願

(全 42 頁)

(21) 出願番号 特願2000-204447 (P2000-204447)

(22) 出願日 平成12年7月6日 (2000. 7. 6)

(31) 優先権主張番号 09/349087

(32) 優先日 平成11年7月8日 (1999. 7. 8)

(33) 優先権主張国 米国 (US)

(71) 出願人 390023157

ノーテル・ネットワークス・リミテッド
NORTEL NETWORKS LIM
ITED

カナダ国 エッチ・2・ワイ 3・ワイ・4
ケベック州 モントリオール セイント
アントワヌ ストリート ウェスト
380 ワールド トレード センター オ
ブ モントリオール エイスフロアー

(74) 代理人 100081721

弁理士 岡田 次生 (外4名)

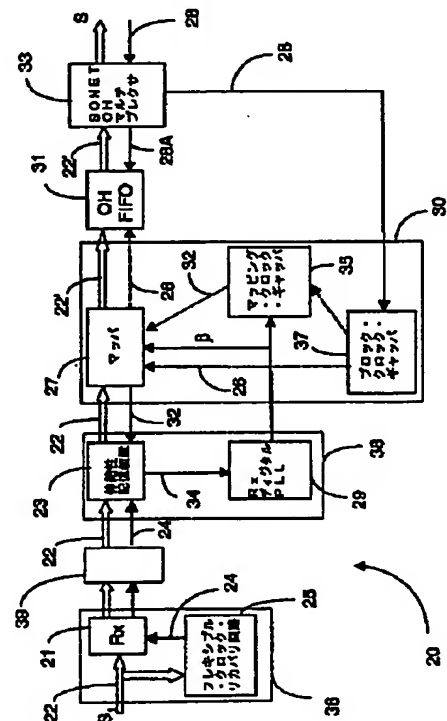
最終頁に続く

(54) 【発明の名称】 S O N E T に任意の信号をマッピングする方法

(57) 【要約】

【課題】 連続フォーマットの信号を S O N E T ネットワークのトリビュタリとしてトランスペアレントに搬送する。

【解決手段】 シンクロナイザ/デシンクロナイザは、任意のレート of 連続フォーマットの信号を、ビットを変化させずジッタとワンダを殆ど加えずに、SONET フレームのような事前選択された単一の共通レートのフレーム内にマッピングする。各フレームは、伝送オーバーヘッド (TOH) ・ビットおよび剰余固定スタフ・ビットを含む一定の数の固定スタフ・ビットを含む。フレームは、また、任意のレートと共通のレートとの位相差に起因する調整可能な数の適応スタフ・ビットを含む。マッピング機能が S O N E T 伝送シェルフのトリビュタリ・ユニット・シェルフにおいて実行され、逆マッピング機能が S O N E T コネクションの遠端において類似の方法で実行される。スタフ・ビットは、フレーム内に均一に拡散される。



【特許請求の範囲】

【請求項1】任意のレートR1の連続デジタル信号を、同期ネットワークを介してトランスペアレントなトリビュタリとして送信する方法であって、前記連続信号の前記任意のレートR1よりも高いレートRの固定長コンテナ信号を選択する段階と、送信サイトで、前記連続信号のビットを前記コンテナ信号のフレームの有効タイムスロットに分散させ、前記フレームに均一に分散された無効タイムスロットにスタッフ・ビットを提供する段階とを含む方法。

【請求項2】前記コンテナ信号が、SONET/SDH信号であり、前記同期ネットワークがSONET/SDHネットワークである請求項1に記載の方法。

【請求項3】前記SONET/SDH信号が、さらに同期トリビュタリを含む請求項2に記載の方法。

【請求項4】前記SONET/SDH信号が、複数のトランスペアレントなトリビュタリを含む請求項2に記載の方法。

【請求項5】前記無効タイムスロットが、固定スタッフ・ビットと適応スタッフ・ビットの一方を含む請求項1に記載の方法。

【請求項6】前記分散させる段階が、データ・ビットの連続ストリームを受け取り、前記任意のレートR1と前記レートRの位相差を決定する段階と、前記位相差に基づいて、前記連続ストリームに、前記フレーム内に前記固定スタッフ・ビットを収容する一定の数のタイムスロットおよび前記フレーム内に前記適応スタッフ・ビットを収容するための調整可能な数のタイムスロットを加える段階と、を含む請求項5に記載の方法。

【請求項7】前記調整可能な数が、前記一定の数よりも実質的に大きい請求項6に記載の方法。

【請求項8】前記一定の数が、伝送オーバーヘッドTOHタイムスロットと、剰余固定スタッフ・ビット・タイムスロットとを含む請求項6に記載の方法。

【請求項9】前記TOHタイムスロットに、保守、運用、管理および設備提供情報を提供する段階をさらに含む請求項8に記載の方法。

【請求項10】前記加える段階が、前記フレームを、いくつかの等しいサイズのデータ・ブロックと、前記一定の数のタイムスロットに分割する段階と、各ブロックごとに、固定スタッフ・ビットの数を決定し、前記ブロック内に前記固定スタッフ・ビットを均一に分散させる段階と、前記調整可能な数を示す制御関数 β を決定する段階と、前記制御関数に基づいて前記固定スタッフ・ビットと前記適応スタッフ・ビットを次のブロック内に均一にマッピングする段階と、を含む請求項6に記載の方法。

【請求項11】前記マッピングする段階が、前記ブロック内のタイムスロットを識別するカウンタCを提供する段階と、

前記制御関数 β の逆の2進ビット α を定義する段階と、前記カウンタCのビット遷移デルタを計算する段階と、関数Valid(C, β)が偽のときに、前記カウンタCによって識別されるタイムスロットが無効タイムスロットであるかどうかを決定する段階と、

適応スタッフ・ビットを前記無効タイムスロット内に提供する段階と、

を含む請求項10に記載の方法。

【請求項12】前記フレームの前記有効タイムスロットから前記連続信号のデータ・ビットを取り出すことによって、受信サイトにおいて前記同期信号から前記連続信号を復元する段階をさらに含む請求項1に記載の方法。

【請求項13】同期ネットワークを介してトランスペアレントなトリビュタリ信号として伝送するために任意のレートの連続フォーマットの信号をマッピングするシンクロナイザであって、

前記連続フォーマットの信号を受け取り、データ・ビットのストリームと前記任意のレートを示すデータ・クロックとを復元するデータ・リカバリ・ユニットと、

前記データ・ビットのストリームを受け取り、前記任意のレートと前記トリビュタリのフレームのレートとの位相差を決定し、制御関数 β を生成するレシーバ・バッファ・ユニットと、

前記レシーバ・バッファ・ユニットから前記データ・ビット・ストリームをマッピング・クロック・レートで取り出し、スタッフ・ビットとデータ・ビットのカウントを、前記制御関数 β にしたがってブロック・クロック・レートで前記フレーム内に一様に分散させるマッピング・ユニットと、

を含むシンクロナイザ。

【請求項14】前記レシーバ・バッファ・ユニットが、ある量の前記ストリームのデータ・ビットを前記データ・クロックで一時的に記憶し、前記データ・ビットを前記ブロック・クロック・レートで前記マッピング・ユニットに提供する伸縮性記憶装置と、

前記任意のレートと前記マッピング・クロックとの位相差を決定し、前記制御関数 β を提供するデジタルPLLと、

を含む請求項13に記載のシンクロナイザ。

【請求項15】前記データ・リカバリ・ユニットが、前記任意のレートを検出するための周波数敏捷性PLLと、前記データ・クロックを使用して前記データ・ビットを検出するレシーバとを含む請求項13に記載のシンクロナイザ。

【請求項16】前記マッピング・ユニットが、前記同期フレームのレートを示すクロックを受け取り、前記同期フレームのすべてのタイムスロットを与えるプ

ロック・レートの前記ブロック・クロック、および固定スタッフ・ビットを収容する一定の数のタイムスロットを与えるギャップを提供するブロック・クロック・ギャップと、

前記ブロック・クロックと前記制御信号 β を受け取り、前記同期フレームのすべてのタイムスロットを与えるマッピング・レートのマッピング・クロック、および前記フレーム内の適応スタッフ・ビットを収容する調整可能な数のタイムスロットを与えるギャップを提供するマッピング・クロック・ギャップと、

前記ブロック・クロックと前記マッピング・クロックを受け取り、それに従って前記フレーム内の前記データ・ビットのストリームをマッピングするマップと、を含む請求項13に記載のシンクロナイザ。

【請求項17】前記同期ネットワーク内で前記フレームをシームレスに伝送するために複数の伝送オーバーヘッドTOHタイムスロットを再配列するためのレシーバOH FIFOをさらに含む請求項13に記載のシンクロナイザ。

【請求項18】前記TOHタイムスロット内に運用、管理、保守および設備提供データを追加するオーバーヘッド・マルチプレクサをさらに含む請求項17に記載のシンクロナイザ。

【請求項19】トランスペアレントなトリビュタリとして同期ネットワークを介して受け取った任意のレートの連続フォーマットの信号を逆マッピングするデシンクロナイザであって、

ブロック・クロック・レートで前記トリビュタリのフレームを受け取り、制御関数 β を受け取り、データ・ビットのストリームをマッピング・クロック・レートで取り出す一方、前記制御関数 β に従ってスタッフ・ビットを除外する逆マッピング・ユニットと、

前記データ・ビットを受け取り、前記任意のレートと前記フレームのレートとの位相差を決定するトランスミッタ・バッファ・ユニットと、

前記データ・ビットを受け取り、前記位相差によって制御されたデータ・レートで前記連続フォーマットの信号を送信するためのデータ送信ユニットと、を含むデシンクロナイザ。

【請求項20】前記制御関数 β が、前記フレーム内で受け取られる請求項19に記載のデシンクロナイザ。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、連続フォーマットを有するデータ信号のトランスペアレントな（透過性の）伝送を対象とし、詳細には、任意の連続信号をSONETフレーム内にマッピングする方法を対象とする。

【0002】

【従来の技術】顧客が要求するサービスを迅速に提供することは、ネットワークの重要な機能である。そのよう

なサービスのビット・レートのレンジは大きくなる可能性があり、実際に、ネットワーク装置が導入されるときにサービスとそのビット・レートが定義されないこともある。したがって、任意のビット・レートのサービスを迅速に提供することは重要な機能である。

【0003】データの伝送フォーマットは、SONETやその他の連続フォーマットとパースト・フォーマットとに分けることができる。パースト・フォーマットには連続クロックがなく、そのような信号の送信には、パースト間に所定の位相関係を必要としない。一方、連続フォーマットのクロックの位相は、通常状態で連続性を有し、クロックの周波数が制限される。そのような制限の例は、 $\pm 20 \text{ ppm}$ （ビット・レートの ppm (parts per million)）と $\pm 100 \text{ ppm}$ である。

【0004】光ファイバ・ネットワークにおける主流の信号フォーマットは、北アメリカでは同期規格SONETに従い、その他の地域ではSDHに従う。本明細書では、SONETは、SDHを含むように定義される。SONETは、信号の多重化、追加（adding）および引き込み（drop）、ならびに一般伝送を可能にする。サービスに関して、SONETネットワークによって容易に伝送することができることは、ネットワーク提供者が導入されたSONET適合装置の大きな基盤を利用できるようにするという点で重要な特性である。

【0005】SONETは、ATM、SMDS、フレーム・リレー、T1、E1などの伝送サービスを提供することができる物理的な搬送技術である。また、SONETの運用、管理、保守および設備提供（provisioning）

（OAM&P）機能は、バック・ツー・バック多重化の量を少なくすることができ、さらに重要なことに、ネットワーク提供者が、ネットワークの運営コストを削減することができる。

【0006】SONET規格ANSI T1.105とBellcore GR-253-COREは、物理インタフェース、光信号キャリア（OC）として知られる光学回線速度、フレーム・フォーマット、およびOAM&Pプロトコルを定義する。SONETネットワークの周辺部において光学的／電気的変換が行われ、そこで、光信号が、光信号の等価物である同期伝送信号（STS）と呼ばれる標準電気フォーマットに変換される。すなわち、STS信号は、搬送するSTSにしたがって定義された各光キャリアによって搬送される。したがって、信号STS-192は、光信号OC-192によって搬送される。

【0007】STS-1フレームは、90列×9行のバイトからなり、フレーム長は、125マイクロ秒である。フレームは、3列×9行のバイトを占める伝送オーバーヘッド（TOH）と、87列×9行のバイトを占める同期ペイロード・エンベロープ（SPE）とを含む。SPEの第1列は、パス・オーバーヘッド・バイトで占めら

10

20

30

40

50

れる。

【0008】したがって、STS-1は、51.840 Mb/秒のビット・レートをもつ。低い方のレートは、STS-1のサブセットであり、DS3より低いレートで伝送するかもしれない仮想トリビュタリ(VT)として知られる。高い方のレートSTS-Nは、SONET追加/引き込み・マルチプレクサを使用して低い方のレートのトリビュタリ(tributary, 支流)を多重化することによって構成される。ここで、N=1, 3, 12, . . . 192またはそれ以上である。信号STS-Nは、N個のSTS-1信号をインターリーブすることによって得られる。たとえば、STS-192は、それぞれ別々に見え、エンベロープ内に別々に並べられた192個のSTS-1トリビュタリからなる。個々のトリビュタリは、それぞれ異なる宛先を有する異なるペイロードを搬送することができる。

【0009】STS-Nは、個々のトリビュタリの全部でN個のTOHからなるTOHと、トリビュタリの全部でN個のSPEからなりそれぞれ自分のPOH(パス・オーバーヘッド)を有するSPEとを有する。

【0010】より高い速度で動作するいくつかのサービスは、STS-Nc信号(連結(concatenation)のc)で送信される。STS-Nc信号内にSTS-1と一緒に維持される。STS-Nc信号のエンベロープ全体は、N個の別々のエントリとしてではなく単一のエントリとして経路指定され、多重化され、伝送される。Nの構成要素のためのTOHとSPEの始まりとは、すべての構成要素が同じソースによって生成されるため、同じくロックにすべて合わされる。連続する信号における最初のSTS-1は、STS-Ncに必要とされる1組のPOHを搬送する。

【0011】あるレートまたはフォーマットを別のレートまたはフォーマットにマッピングする方法は周知である。Bellcore TR-0253は、SONETへの共通非同期伝送フォーマット(DS0, DS1, DS2, DS3など)の標準的なマッピングについて詳細に説明している。これと類似のマッピングが、SDHへのETSI階層マッピングのために定義される。光伝送装置は、ある独自のフォーマットを別のフォーマットにマッピングした。たとえば、FD-565は、標準フォーマットDS3だけでなくNortelの独自フォーマットFD-135を搬送することができる。

【0012】しかしながら、標準または独自の機構は、フォーマットに固有のハードウェアにより、きわめて特有の組の信号の伝送を可能にする。そのようなマッピング方法を使用して、標準と大きく異なるレートをマッピングすることはできない。さらに、そのようなマッピングはそれぞれ、特定のフォーマットと特定のビット・レートに関して、たとえば±20ppmの許容範囲で正確に調整される。信号は、たとえばDS3と1%でも異なる

るビット・レートを有する場合は、SONET内で伝送することができない。さらに、各種の信号のマッピングを行うためには、一般に、異なるハードウェア・ユニットが必要である。

【0013】前述の問題の解決策は、任意の連続信号に「ラッパ(wrapper)」を加えることである。得られる信号のレートは、ラップされる(包まれる)信号の関数である。すなわち、レートXの信号に1Mb/秒のラッパが加えられると、レートX+1Mb/秒を有するフォーマットが生成される。この変化は、Xの割合を高める。たとえば、共通の伝送路符号化8B/10Bは、Xの112.5%のレートを有するフォーマットを作成する。したがって、「ラッパ」法は、任意の入力に対し事前に定義された一定ビット・レートを有するフォーマットを生成しない。一般に、得られた信号は、時分割多重化して高速ネットワーク上で伝送することができない。

【0014】

【発明が解決しようとする課題】米国特許第5,784,594号(Beatty)は、任意の信号が、必要な数のフレームにマッピングされ、残りのフレームが空のままにされる「TDMラッパ」フォーマットを提案している。しかしながら、この方法は、ビットを送る適切なタイムスロットを待っている間ビットを保持するために変換方向ごとにきわめて大きいメモリを必要とする。その結果、このフォーマットは、高速の信号で実現するためにはコストがかかる。

【0015】パケットまたはセル・ベースのフォーマットは、任意の入力ストリームをSONETとSDHにマッピングする。これらの方法は、パケット・システムには適しているが、「1つのサイズがすべてに合う」マッピング方法が使用されるため、ほとんどの連続信号フォーマットのジッタ要件やワンド要件を満たさない。入力信号のクロック位相情報は、そのような方法において完全に削除され、したがって送信することができない。

【0016】米国特許出願第09/307812号(Solheimらによる1999年5月10日出願されNortel Networks Corporationに譲渡された「Protocol Independent sub-rate device」と題する出願)は、異なるタイプのクライアント(IP, ATM, SONET, イーサネット(登録商標)など)と一緒に伝送する方法を開示している。前記出願は、任意のレートおよびフォーマットの低速(サブレート)チャネルを単一の高速チャネルに時分割多重化し、次にそのチャネルをシステムの遠端において多重分離する方法を開示している。任意の所与のサブレート・チャネルに割り当てられた帯域幅部分を、ハードウェアやソフトウェアに変更を加えることなく提供することができる。これにより、キャリアによるそのようなサービスの提供がきわめて容易になり高速化する。新しいプロトコルによるトリビュタリにも対処することができ、そ

のような新しいプロトコルのサポートのための送出力が大幅に高速化される。

【0017】低タイミング・ジッタかつ低コストで信号を復元できるように任意の信号をSONETにマッピングする効率的な方法および装置の必要性が残っている。

【0018】

【課題を解決するための手段】本発明の目的は、連続フォーマットを有する任意の信号をSONETフレーム内にマッピングすることである。これにより、SONETネットワーク内で規格に合う任意のフォーマットをトランスペアレントに伝送することができる。

【0019】したがって、本発明は、任意のレートR1の連続デジタル信号をトランスペアレントなトリビュタリとして同期ネットワークを介して送信し、連続信号の任意のレートR1よりも高いレートRの固定長コンテナ信号を選択し、送信サイトにおいて、連続信号のビットをコンテナ信号のフレームの有効タイムスロットに分散させ、フレームに均一に分散された無効タイムスロットにスタッフ・ビットを提供する方法を含む。

【0020】本発明は、さらに、同期ネットワークを介してトランスペアレントなトリビュタリ信号として伝送するために任意のレートの連続フォーマットの信号をマッピングするために、連続フォーマットの信号を受け取って、データ・ビットのストリームと任意のレートを示すデータ・クロックとを復元するデータ・リカバリ・ユニットと、データ・ビットのストリームを受け取り、任意のレートとトリビュタリのフレームのレートとの位相差を決定し、制御関数 β を生成するレシーバ・バッファ・ユニットと、レシーバ・バッファ・ユニットからマッピング・クロック・レートでデータ・ビットのストリームを取り出し、スタッフ・ビットとデータ・ビットのカウントを、制御関数 β に従ってブロック・クロック・レートでフレーム内に均一に分散させるマッピング・ユニットとを含むシンクロナイザを含む。

【0021】本発明のもう1つの態様によれば、同期ネットワークを介してトランスペアレントなトリビュタリ信号として受け取った任意のレートの連続フォーマットの信号を逆マッピングするために、ブロック・クロック・レートでトリビュタリのフレームを受け取り、制御関数 β を受け取り、データ・ビットのストリームをマッピング・クロック・レートで取り出す一方、制御関数 β に従ってスタッフ・ビットを除外する逆マッピング・ユニットと、データ・ビットを受け取り、任意のレートとフレームのレートとの位相差を決定するトランスミッタ・バッファ・ユニットと、データ・ビットを受け取り、位相差によって制御されたデータ・レートで連続フォーマットの信号を送信するデータ送信ユニットとを含むシンクロナイザを提供する。

【0022】本発明によるマッピングの方法は、同じ形式または異なる形式のトリビュタリをトランスペアレン

トに伝送するためにSONETなどの一般的な技術を使用可能にするため、有利である。この新規のマッピングを使用することによって、ビットを変化させることなくほとんどのすべての連続フォーマットを伝送することができる。本発明のもう1つの利点は、この方法によって加えられるジッタまたはワンドが最小であることである。

【0023】本発明によるシンクロナイザ/デシンクロナイザは、ジッタの許容と生成の仕様が、ユニット内に設計されたきわめて収容力の高いレンジに適合する限り、設計時にフォーマットが分かっている信号を処理する。これは、トリビュタリ・ソフトウェアによって実行中にデザインされる独特なマッピングであり、遠端にある対応するトリビュタリにチャンネル内で送られる。

【0024】本発明の以上その他の目的、特徴および利点は、添付図面に示したような好ましい実施形態に関する以下のより特定の説明から明らかになるであろう。

【0025】

【発明の実施の形態】本発明によるマッピング・システムは、指定された最大容量以下の一定回線速度を有するデジタル信号を、提供されたサイズのSONETエンベロープ内にマッピングする。SONET伝送シェルフのトリビュタリ・ユニットでマッピング機能を実行することができ、SONETコネクションの遠端にある類似のユニットで逆マッピング機能（デマッピングとも呼ばれる）を実行することができる。

【0026】図1Aは、SONETネットワーク上で複数のサービスをトランスペアレントに伝送する本発明によるマッピング・システムを備えた例示的な伝送システムのブロック図を示す。簡略化するため、この図では、矢印で示したような単一方向の伝送だけを示す。

【0027】信号S1, ..., Sj, ..., Snは、SONETネットワーク7を介して、2つのサイトAとBの間で伝送され、SONET信号Sとなる。ここで、nは、トリビュタリの数であり、jは、1つのトリビュタリのレンジである。信号S1~Snは、連続フォーマットのデジタル信号であり、ノードAおよびBにおいてSONET信号Sのトリビュタリとして扱われる。また、各信号SjのレートをRjで表し、信号SのレートをRで表す。信号Sjは、同じタイプまたは異なるタイプのサービスを搬送することができる。各トリビュタリ・レシーバ1~1nは、それぞれの連続フォーマットの信号S1~Snのデータ・ビットを復元する。ノードAは、1つまたは複数のシンクロナイザ201~20nを備え、各シンクロナイザ20jは、相当するトリビュタリ信号Sjのデータ・ビットを適切なサイズのSONETエンベロープにマッピングする。

【0028】いくつかの伝送ノード間で連続信号をトランスペアレントに搬送するフレームのサイズは、ソフトウェアで選択され、大きいレンジの連続フォーマットの

信号に十分な帯域幅使用量を考慮して提供される。たとえば、エンベロープに $n \times \text{STS}-12$ が使用される場合、 n は、高速のシンクロナイザの場合は $4 \sim 20$ であり、中速のシンクロナイザの場合は $1 \sim 5$ である。これにより、大きいネットワーク容量が無駄になるのが防止される。

【0029】各信号が、それぞれのSONETエンベロープ内にマップされた後、トリビュタリは、SONETマルチプレクサ3によって高レート信号Sに多重化され、それが、SONETトランスミッタ5によって光ネットワーク7を介してサイトBの方に送られる。

【0030】逆の操作が、サイトBにおいて行われる。すなわち、光レシーバ9が、信号Sのデータを復元し、デマルチプレクサ3' がその信号を分離し、それを各デシンクロナイザ40jは、トリビュタリ・トランスミッタ11jに提供される信号Sjと関連したそれぞれのフォーマットでビットを再配列する。各トリビュタリ・トランスミッタ11-11nは、各信号S1-Snを、関連したトリビュタリ・ネットワーク上に送り出すか、関連したエンド・ユーザに送り出す。

【0031】次に、基礎的な概念と実現性を示すために、信号STS-192cについてマッピング・アルゴリズムの例を示す。他のエンベロープを使用することもでき、本発明は、信号STS-192cに制限されない。

【0032】図1Bは、TOH2とSTS-192c SPE（同期ペイロード・エンベロープ）4を含むSTS-192cフレーム1を示す。ペイロードは、 $192 \times 87 \times 9 \times 8 = 1,202,688$ ビットを含む。

【0033】ここでは、ブロック10-jが、後で説明するようなデータ・ビット、固定スタッフ・ビットおよび適応スタッフ・ビットを含む1056ビット・フィールドとして定義される。STS-192 SPEは、灰色に示され参照番号8で示された領域を占める1138のブロック10-1~10-K（ここで、 $K=1138$ ）を収容することができる。ブロック・フィールド8は、1,201,728ビットを有する。エンベロープ4内の残りの960ビットは、POHビット6（ $9 \times 8 = 72$ ビット）と、888ビットの剰余フィールド14からなる。フィールド6および14のビットの数は、SONETフレーム1内にマップされる連続フォーマットの信号のレートR1に関係なく不変である。したがって、これらのビットは、以下において固定スタッフ・ビットと呼ばれる。

【0034】一方、ブロック・フィールド8を埋めるのに必要なスタッフ・ビットの数は、連続フォーマットの信号S1のレートR1の関数を変化させる。これらのスタッフ・ビットは、本明細書において、適応スタッフ・ビットと呼ばれる。

【0035】本発明によれば、信号S1のデータ・ビットは、固定スタッフ・ビットと適応スタッフ・ビットが均等に分散されたフレーム1内にマッピングされる。そのようなスタッフ・ビットは、レートR1があらかじめ分からない場合があるため、実行中に、各ブロックに均等に分散される。したがって、シンクロナイザは、現行ブロックのデータ・ビットがマッピングされるときに蓄積された位相情報に基づいて、データ・ビットの場所である有効位置と、次のブロックのスタッフ・ビットの場所である無効位置を定義する。さらにまた、シンクロナイザは、実際のマッピング時にオーバーヘッドを均等に分散させるが、マッピング動作後にそれをSONET規格に従って提供されるタイムスロットに再編成し、その結果、フレームがSONET装置によって認識される。遠端において、シンクロナイザは、固定スタッフ・ビットと適応スタッフ・ビットを吸収することによって逆の動作を行い、それによりデータ・ビットを逆マッピングしてS1を再生することができる。

【0036】図1Bは、フレームの構造を直観的に示すが、本発明によれば、マッピング・アルゴリズムが、固定スタッフ・ビットと適応スタッフ・ビットをフレーム1内に均等に分散させることに注意されたい。以上の計算は、STS-192cフレームに適用することができるが、類似の事が他のSONET信号にも当てはまることに注意されたい。

【0037】各ブロックのビットは、図1Cに示したように割り振られる。ブロック10-1は、データ用に1023（ $2^{10}-1$ ）のビットを有するデータ・フィールド17と、16ビットを有する制御フィールド13と、将来使用するための17ビットを有するスペア・フィールド15とを含む。

【0038】フィールド17の1023ビットは、データをSTS-192cフレーム内伝送するための9,313,392Mbps（ $1023 \times 1138 \times 8000$ ）のビット・レートを提供する。フレームのサイズは、一定の用途のために提供され固定され、すなわち、任意のレートのトリビュタリが、同じサイズのフレームにマッピングされる。マッピング技術は、トリビュタリごとに異なるフレームが使用されるのではなく、任意のトリビュタリに適応する。トリビュタリが、9,313,392Mbpsより低いレートを有する場合は、トリビュタリをSTS-192cに合うように調整し、データ・フィールド8の多くのビットをスタッフ・ビットに変化させなければならない。図1Cは、フィールド17内の可変サイズvのフィールド19を示し、そのサイズは、マッピング中に、信号S1のクロックと信号Sのクロックの位相を比較することによって決定される。

【0039】フィールド13は、10ビットの制御関数βを含む。βのサイズは、ブロックのサイズに従って、後で述べる適応スタッフ・アルゴリズムにより次

のブロック内の有効ビットの位置を一意に決定するように選択される。10ビットという数によって、1024個の値を想定することが可能であり、この数はブロックの有効ビットの数よりも1大きい。 β の値は、また、次のブロックにおける有効ビットの数を与える。単一ビット誤り訂正と複数誤り検出には、フィールド13の追加の6ビットが必要である。

【0040】複数誤り検出の場合、前のブロックからの β が、デフォルトとして、最小のPLL過渡事象でダウンストリームを高速でリフレームするのに使用される。フィールド15および19のビットは、ブロック内に分散される。

*

 α の決定

β										α									
β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}	α_{10}	α_9	α_8	α_7	α_6	α_5	α_4	α_3	α_2	α_1
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	1
0	1	1	1	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0

【0044】 α は、また、ブロックごとに決定され、 β の場合と同じように、 α の値は、隣り合ったブロック間では異なるが、ブロック内では変化しない。

【0045】また、ここで、カウンタCと値Dが定義される。Cは、1ブロック内のビットのカウンタであり、10ビットの2進数によって表される。Cは、1から1023まで増加し、それによりブロック内のビットが占めるタイムスロットを識別する。

【0046】Dは、Cのビット遷移デルタであり、ちょうど1つのビット・セットによる10ビット2進数によって表される。このセット・ビットは、カウンタCが1 ※

所与の値Cに関するDの算出

C _{1,2,...,n} ブロック内のビットのカウンタ										D _{1,2,...,n} Cのビット遷移デルタ									
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_n	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_n
$C=1$	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
$C=5$	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
$C=1$	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
$C=2$	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1

【0050】ブロックの有効データ間にスタッフ・ビットをある程度均一に分散させるために、本発明による適応スタッフィング・アルゴリズムは、Dにおけるビットが α においてもセットされるときにビットが有効であることを述べている。

★

Valid(C, β), if any bit of ($\alpha_{1,2,...,n}$ AND $D_{1,2,...,n}$) is non zero

【0053】有効ビットは、データに割り当てられたビットに対応し、したがって無効ビットは、スタッフ・ビットに対応する。表3は、 β のサイズが3ビットの場合に、アルゴリズムが7ビットのブロックにどのように機能するかの簡単な例を示す。アルゴリズムは、1023ビットと10ビット β のブロックに同じようにはたらくことが分かるが、本明細書では、完全なスタッフィング

* 【0041】すべてのブロックが同じ数の適応スタッフ・ビットを有するとは限らないので、 β の値は、隣り合ったブロックで異なることがあるが、各ブロック内で β は一定のままである。

【0042】適応スタッフィング・アルゴリズムは、 α で表された、 β と逆の2進ビットを定義する。すなわち、 β の最上位ビットが、 α の最下位ビットになり、同様に、 α の最下位ビットが、 β の最上位ビットになる。表1は、この変換を例として示す。

【0043】

【表1】

※ビット進むときに生ずる0から1への遷移の位置にある。Dの各ビットは、ブール関数を使用して、次の式に従って、カウンタCのレンジnとn-1のビットによって与えられる。

【0047】

【数1】 $D_n = C_n \text{ AND NOT } (C-1)_n$

【0048】表2は、所与の値Cに関してDがとる値の例を示す。

【0049】

【表2】

★ 【0051】これは、ブロックのC番目のビットについて、次の式2のように示すことができる。

【0052】

【数2】

・シーケンスについて詳述することは实际的でない。

【0054】表3の項目は、2進関数Valid(C, β)の結果である。列は、 β で与えられたようなブロック内の特定数の有効ビットにデータ・ビットとスタッフ・ビットがどのように分散されているかを示す。

【0055】Valid(C)が真の場合のCの各値に関して、有効データ・ビットが、Cで識別されるタイム

スロット内にあり、Valid (C) の真でない各値に関して、スタッフ・ビットがタイムスロットに入れられる。この方式を使用することにより、無効スタッフィング・

*グ・ビットが、フレーム全体にほぼ均一に広がる。

【0056】

【表3】

3ビットβの場合の7ビット・ブロックのフレキシブル・マッピングの例

	β	000	001	010	011	100	101	110	111
α	000	100	010	110	001	101	011	111	000
C	β	Valid(C,0)	Valid(C,1)	Valid(C,2)	Valid(C,3)	Valid(C,4)	Valid(C,5)	Valid(C,6)	Valid(C,7)
1	000	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data
2	010	011	Stuff	Stuff	Data	Data	Stuff	Stuff	Data
3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data
4	100	101	Stuff	Data	Stuff	Data	Stuff	Data	Stuff
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data
6	110	011	Stuff	Stuff	Data	Data	Stuff	Stuff	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data

【0057】この例では、カウンタCは、1から7までカウントし、Dの値が、各列においてCのすべての値に関して求められる。次に、Dの各値は、Cの増加に伴い、αと比較される。また、Dのセット・ビットがαにおいてセットされる場合は、ブロック内の対応するC番目のビットがデータ・ビットになる。Dのセット・ビットがαにおいてセットされない場合、ブロック内の対応するC番目のビットはスタッフ・ビットになる。

【0058】例としてビット・レートR1/Rが有効能力の5/7のブロックの場合は、βが2進数5（101）であり、βの逆数の2進ビットαも5（101）であることを意味する。ブロック内のデータ・ビットとスタッフ・ビットの順序は、列ごとのValid (C, 5)と同じであり、次の通りである。

【0059】Data, Stuff, Data, Data, Data, Stuff, Dataまた、表3に関して、Valid (C, 5) の場合、5であるβは有効ビットの数でもあり、無効ビットがフレーム全体にほとんど均一に広がっていることは明らかである。

【0060】図4および図5は、βがブロック間で少し異なる5つの連続ブロックのスタッフィング・シーケンスを連続して詳細に示す図である。図4および図5は、一点鎖線の部分においてつながる。この表において、カウンタCは、1から7までカウントし、2つの連続するブロックは、分かり易くするために異なる地色（灰色と白）で示されている。

【0061】前に示したように、βは、各ブロックごとにセットされるが、スタッフ・ビットの総数とブロックの数の比が整数でない場合もあるため、次のブロックと異なることがある。次の例を検討する。ブロック1（灰色）のβは5である。ブロック2（黒）のβは6である。ブロック3（灰色）のβは5である。ブロック4（黒）のβは5である。ブロック5（灰色）のβは6で

ある。

【0062】この場合、データ・ビット間のスタッフ・ビットの広がり、は、図6に例示される通りである。ここで、Dはデータを表し、Sはスタッフを表す。上記のDとSの混乱を避けるために、これらは通常の文字で記述され、信号Sと、Cのビット遷移Dにはイタリック体で使用される。

【0063】この場合も、図4および図5から、スタッフ・ビットが、βのわずかな変動があるものの、データ・ビット間にある程度均一に広がっていることが明らかである。

【0064】図2は、トリビュタリ・シンクロナイザ20のブロック図を示す。任意のレートで受け取ったデータでSONET SPEを満たすことによって、前述のような透明性が得られる。データ経路は、幅の広い矢印および参照数字22と22' を使って示される。連続フォーマットおよびレートR1を有する信号S1が、データ・リカバリ・ユニット36によって検出される。次に、データ・ビットは、充填制御ボックス38、マッピング・ユニット30、レシーバ・オーバーヘッドFIFO（先入れ先出し）31、およびオーバーヘッド・マルチプレクサ33を通る。このとき、シンクロナイザ20から出力された信号は、SONETフレーム内にある。信号Sが、それぞれのOAM&P情報と供にSONET型オーバーヘッド（TOHとPOH）を有し、同期ペイロード内へのビットの配置が、SONETの規格と異なるマッピング・アルゴリズムに従うことを理解されたい。

【0065】シンクロナイザ20が、異なる4つのクロック、すなわちデータ・クロック24、ブロック・クロック26、マッピング・クロック32、ならびにSONETクロック28および28Aを操作する。クロック28は、STS-192のレートを有し、クロック28Aは、フレームのレートを有する。データ・クロック24

(レートR1)は、レシーバ21とフレキシブル・クロック・リカバリ回路25を含むデータ・リカバリ・ユニット36によって入力データから復元される。フレキシブル・クロック・リカバリ回路25は、広い連続した範囲のビット・レートにわたるクロック・リカバリが可能である。そのような回路の例は、1998年12月22日に出願され、Northern Telecom Limitedに譲渡された、Habelらによる「Apparatus and Method for Versatile Digital Communication」と題する同時係属米国特許出願09/218053に開示されている。この出願は、参照により本明細書に組み込まれる。

【0066】図2において点線で示したオフ・ライン・フレーム39が、ある組の既知の信号フォーマットを認識することができ、フレームとBER性能情報が報告される。また、マッピング効率を高めるために、回線符号化をレシーバにおいていくつかの信号から除去し、トランスミッタにおいて加えることができる、これらのオプションは、特定タイプのサービスに依存し、したがってここではさらに詳しく考察しない。

【0067】レシーバ・バッファ・ユニット38は、伸縮性記憶装置23とレシーバ・デジタルPLL29を含む。データ・クロック24は、マッピング・クロック32の制御下で空にされる伸縮性記憶装置23へのデータの入力をクロックするために使用される。マッピング・クロック32は、STS-192クロック28から得られたギャップド・クロックである。このクロックは、ブロック・クロックにおけるギャップの他に、マッピング・アルゴリズムによって決定されるような適切な位相時に中断される。この方法において、データ22は、マ

【0068】ペイロード・フィールド4に、データ・ビットが必要な容量まで連続的に満たされ、残りの容量がスタッフ・ビットの連続体である場合、伸縮性記憶装置23の容量は、幅広く変化することになり、記憶装置23にある程度大きい深さを必要とする。伸縮性記憶装置23は、スタッフ・ビットがロードされている間に急速に満たされ、トリビュタリ・データ・ビットの連続ストリームがロードされている間に急速に空になる。この状況は、図2の構成において回避され、この場合、伸縮性記憶装置23は、マッピング・クロック32によって実質的に規則正しい間隔で空にされる。

【0069】一方、伸縮性記憶装置23は、トリビュタリからのすべての入力ジッタおよびワンダを十分に吸収できる深さでなければならない。伸縮性記憶装置23の充填が十分に制御されれば、最悪の場合のジッタとワンダが存在してもオーバーフローしたりアンダーフローしないことを保証することができ、またシンクロナイザ20は、ジッタの許容要件を満たす。伸縮性記憶装置23

の最小サイズは、実験的に、256ビットで決定された。

【0070】レシーバ・デジタルPLL29は、マッピング・クロック32を決定する β によって最適な充填を維持するように伸縮性記憶装置を空にする割合を制御する。換言すると、マッピング・クロック32の平均レートは、データ・クロック24の平均レートをたどるように制御され、 β は、それらのクロック間の位相差から得られる。前に示したように、 β は、次のブロックの充填を制御する。この制御は、シンクロナイザ20がSONET規格に従うポイントの調整を必要としないという利点を有する。より正確に言うと、伸縮性記憶装置の充填の制御は、最大トリビュタリ・レートがペイロード・レートを超えない限り、時間によるラインおよびトリビュタリのレート変動(ラインおよびトリビュタリのジッタおよびワンダ)を改善する。

【0071】 β を決定するために、伸縮性記憶装置23への入力が、周期的にサンプリングされ、データ22の位相情報が、PLL29に入力される。たとえば、デジタルPLL29は、24ビットのアキュムレータを含む。ブロックの最初に、マップ27のカウンタCによって与えられた伸縮性記憶装置23の充填は、たとえば50%を基準としてラッチされる。次に、サンプル34の位相は、アキュムレータ内に加えられ、3ビットだけ左にシフトされた位相に加えられる。この和Sの上位10ビットは、 β である。アキュムレータは、ロールオーバーしないようにFFFFFFFFでクリップされ、アナログ出力PLLレンジの低い周波制限を反映するために40000などの最も低い値でクリップされなければならない。その他のデジタルPLLの実施態様も可能である。

【0072】伸縮性記憶装置23が満杯になり始めると、マッピング・クロック32の速度を高めることによって記憶装置を空にするように β が増加される。同様に、記憶装置23が空になり始めると、記憶装置23が充填することができるように β が減らされる。目標の充填率は、50%が好ましい。

【0073】マッピング・ユニット30は、ブロック・クロック・ギャップ37、マッピング・クロック・ギャップ35およびマップ27を含む。

【0074】ブロック・クロック・ギャップ37は、SONET TOHのギャップと規則的サイクルを特徴とするSTS-192クロック28を受け取る。クロック28は、前述の例において、42, 432のギャップがフレーム全体に均一に広がった1SONETフレーム当たり1138(ブロック数) \times 1056(ブロック・サイズ)=1,201,728サイクルを有するブロック・クロック26を生成する。前に示したように、ブロック・クロック26のギャップは、サイズが3 \times 9 \times 8 \times 192のSONETオーバーヘッド、すなわち図1Bのフ

フィールド2と、サイズが960の固定スタッフィング、すなわちフィールド6および14によるものである。ブロック・クロック26は、フィールド8におけるビットの全体の割振りを表す。換言すると、フレームのオーバーヘッド・サイズがSONET規格に従う場合に、ブロック・クロック26は、TOHビット、POHビット、および固定スタッフ・ビットのための空間を維持するために約30個のビットに切断される。

【0075】マッピング・クロック・ギャップ・ブロック35は、ブロック・クロックと同じレートを有するが、前述のように、すべての有効ビット位置におけるパルスによりβの制御下でさらにギャップが開けられ、レートR1とRの間の差に基づいて適応スタッフ・ビットを生じさせる。

【0076】マップ27は、マッピング・クロック32、ブロック・クロック26、および簡単にするために示していない他の補足的クロックを利用し、固定スタッフ・ビットと適応スタッフ・ビットの両方を使用してデータ22の位置を調整する。マッピング・クロック32を使用して、伸縮性記憶装置23からデータ・ビットがマップにクロック・タイミングで引き出される。ブロック・クロック26を使用して、マップ27から、データ・ビット、固定スタッフ・ビットおよび適応スタッフ・ビットがクロック・タイミングで引き出される。マップ27は、本質的にメモリを持たず、伸縮性記憶装置23とFIFO31は、シンクロナイザのすべてのメモリを表す。

【0077】データ、固定スタッフ・ビットおよび適応スタッフ・ビットを含むような参照数字22'で示されたマップ27からの複数のビットは、SONETオーバーヘッドの場所のためにタイムスロットを予約するレシーバ・オーバーヘッドFIFO（先入れ先出し）31にクロック・タイミングで入れられる。次に、ビット22'は、クロック28Aによってクロック・タイミングでFIFO31から引き出され、それによりFIFO31は、各フレームごとに一度同期してリセットされる。FIFO31の深さは、フレームOHがOH MUXにクロック・タイミングで入れられているときにフレームの位相瞬間においてペイロード・ビットだけを記憶するのに十分であれば良い。フレームが、SONETと同じペイロード対OHの比を有する場合、この深さは、 $192 \times 8 \times 9 \times 3$ ビットより大きくなければならず、 $192 \times 8 \times 12 \times 3$ ビットよりも大きいことが好ましい。

【0078】ビット22'は、FIFOブロック31から、SONETオーバーヘッド・マルチプレクサ33にクロック・タイミングで入れられ、そこでSONETオーバーヘッドが、それぞれ空のタイムスロットに加えられ、次に信号が、STS-192として処理される。細い線で示したSONETクロック28および28Aは、通常通り、シェルフの残りの部分に対してロックされる。

【0079】簡単にするために、直列ハードウェア実装について説明する。バイト幅の実装のようなこの種のマッピングの並列実装により、クロック速度を低くすることができる。そのような並列実装は、ジッタを減少させるためにブロック・アライメントを交互にすることがあった。PLLを最適化するために、ハードウェア制御よりもDSP制御の方が大きな自由度を提供する。

【0080】図3は、トランスベアレントな逆方向のシンクロナイザ、すなわちデシンクロナイザ40のトランスミッタ側のブロック図を示す。デシンクロナイザ40は、シンクロナイザ20によって実行されるのとは逆の機能を類似の方法で実行し、類似したブロックを備える。

【0081】SONETオーバーヘッド・デマルチプレクサ53は、シェルフの残りの部分に通常通りロックされるSTS-192クロック28を使用して、信号42'からSONETオーバーヘッドを示す。トランスミッタ・オーバーヘッドFIFO51は、各フレームごとに、クロック28Aと同期してリセットされる。オーバーヘッドFIFO51は、データ・ビットを固定スタッフ・ビットおよび適応スタッフ・ビットと一緒に含むペイロードに受け取ったデータ42'をマップ47に提供するようにオーバーヘッド位置を吸収する。

【0082】ポインタの位置調整を考慮しない場合、トランスミッタOH FIFO51は、シンクロナイザ20のレシーバOH FIFO31と類似の深さを有することがある。たとえば、SONET OHがフレームに使用される場合、必要な深さは、FIFO31の場合と同じように、 $192 \times 8 \times 12 \times 3$ ビットである。したがって、FIFOは、OHがビット・ストリームから多重分離されるフレーム段階の間にトランスミッタOH FIFO51が空にならないように十分なデータ・ビットを記憶することができる。しかしながら、デシンクロナイザにポインタ・アライメントが必要であるため、トランスミッタOH FIFO51は、最悪の場合の一連の正または負のポインタ調整イベントを許容するように深さを大きくしなければならない。

【0083】逆マッピング・ユニット50は、逆マップ47、マッピング・クロック・ギャップ55、およびブロック・クロック・ギャップ57を含む。

【0084】ブロック・クロック・ギャップ57は、STS-192クロック28のギャップを調整してブロック・クロック26を作成する。ブロック・クロック26は、シンクロナイザ20の場合と同じように、1フレーム当たり1,201,728のサイクルを有し、フレーム全体に42,432のギャップが均一に広がる。ギャップは、図1Bのフィールド2,6および14に相当する。換言すると、このクロックは、TOHと固定スタッフ・ビットを拒否する。

【0085】デシンクロナイザのブロック・クロック・

ギャップ57は、また、ポインタ調整のためにギャップを含めたり削除したりする。そのようなギャップは、ポインタ調整による位相ヒットを最少にするために3つのフレームに広げなければならない。

【0086】マッピング・クロック・ギャップ55は、ブロック内のインバンドOHチャネルから読み取ったブロック・クロック26および β を受け取る。ギャップド・クロック26は、 β を使用して、マッピング・クロック32を生成するようにさらにギャップが調整される。マッピング・クロック32は、厳密にトリビュタリ・データ・ビット42がクロック・タイミングで取り出されるようにデータ・ビット42'をギャップ・タイミングで取り出す。

【0087】ビット42は、次に、伸縮性記憶装置43とトランスミッタ・デジタルPLL49を含むトランスミッタ・パッファ・ユニット54によって処理される。トリビュタリ・データ・ビット42は、マッピング・クロック32を使用して出力伸縮性記憶装置43にクロック・タイミングで入れられる。伸縮性記憶装置43は、フレキシブル・クロック・リカバリ回路45から出力されたデータ・クロック24によって空にされる。

【0088】出力伸縮性記憶装置43の位相は、トランスミッタ・デジタルPLLによって周期的にサンプリングされる。サンプル34は、デジタル的に処理され、出力信号がフレキシブル・クロック45に渡され、VCOの電圧が制御される。フレキシブル・クロック回路45は、シンクロナイザのフレキシブル・クロック回路25を備えた類似のタイプであり、データ・クロック24を提供する。

【0089】TxPLL49の帯域幅は、マッピングとポインタ調整によるジッタをフィルタリングできるほど低く、またVCO雑音を抑制できるほど高くなければならない。

【0090】本発明を、特定の実施形態の例に関して説明したが、本発明の意図から逸脱することなくその広い態様において、併記の特許請求の範囲内で当業者が想起するさらに他の修正および改良を行うことができる。

【図5】

1	001	001	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
2	010	010	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
3	011	011	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
4	100	100	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
5	101	101	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
6	110	110	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
7	111	111	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
8	001	001	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
9	010	010	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
10	011	011	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
11	100	100	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
12	101	101	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
13	110	110	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
14	111	111	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
15	001	001	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
16	010	010	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
17	011	011	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
18	100	100	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
19	101	101	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
20	110	110	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff
21	111	111	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff	Stuff

【図面の簡単な説明】

【図1】 本発明を例示する図であり、図1Aは本発明によるマッピング・システムを備えた通信ネットワークのブロック図である。図1Bは本発明の実施形態によるブロックを示すOC-192cフレームの図である。図1Cはブロックの構造の例を示す図である。

【図2】 本発明の実施形態によるシンクロナイザのブロック図である。

【図3】 本発明の実施形態によるデシンクロナイザのブロック図である。

【図4】 適応スタッフィング・アルゴリズムを例示する表を示す図である。

【図5】 図5に続く、適応スタッフィング・アルゴリズムを例示する表を示す図である。

【図6】 スタッフ・ビットの広がり为例示する図である。

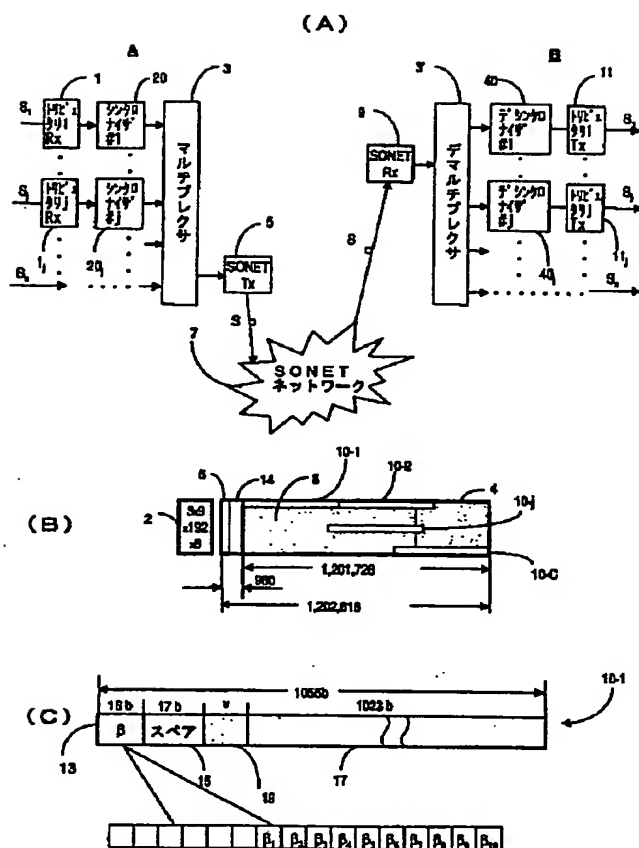
【符号の説明】

- 20 シンクロナイザ
- 21 レシーバ
- 22 データ
- 23 伸縮性記憶装置
- 24 データ・クロック
- 25 フレキシブル・クロック・リカバリ回路
- 26 ギャップド・クロック
- 27 マップ
- 28 STS-192クロック
- 30 マッピング・ユニット
- 31 FIFO
- 32 マッピング・クロック
- 33 オーバヘッド・マルチプレクサ
- 34 サンプル
- 35 マッピング・クロック・ギャップ・ブロック
- 36 データ・リカバリ・ユニット
- 37 ブロック・クロック・ギャップ
- 38 レシーバ・パッファ・ユニット
- 39 オフ・ライン・フレーム

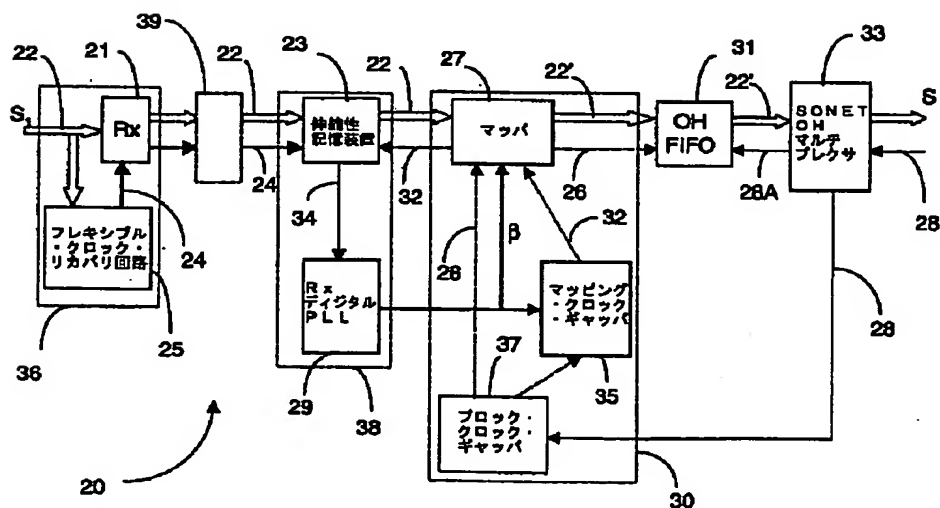
【図6】

灰色 黒 灰色 黒 灰色
DSDDSDS DDDSDDD DSDDSDS DSDDSDS DDDSDDD

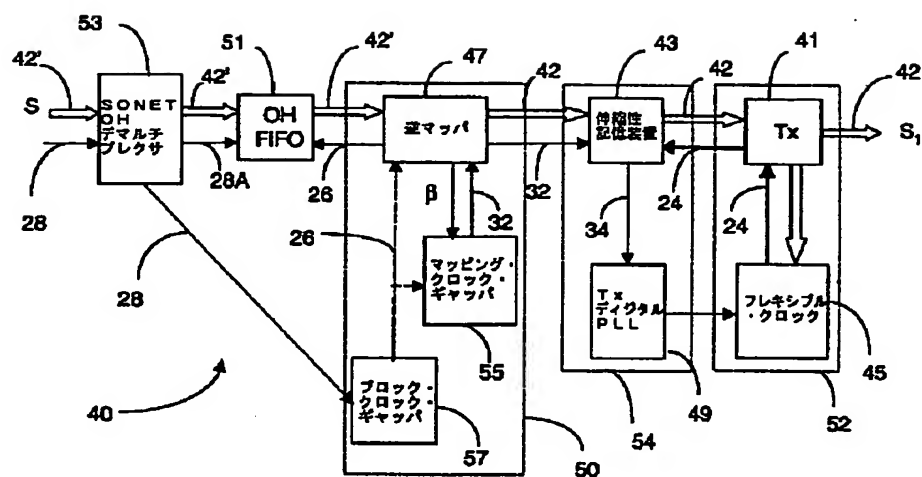
【图 4】



【図 2】



【図3】



フロントページの続き

(73) 発明者 キム・ビー・ロバーツ

カナダ、ケー2アール、1シー6、オンタ

リオ、ネピアン、ミッション・イン・グロ

ーヴ 10

【外国語明細書】

1. Title of Invention

MAPPING ARBITRARY SIGNALS INTO SONET

2. Claims

1. A method for transmitting a continuous digital signal of an arbitrary rate R1 over a synchronous network as a transparent tributary, comprising:
 - selecting a fixed length container signal of a rate R, where R is higher than said arbitrary rate R1 of said continuous signal; and
 - at a transmit site, distributing the bits of said continuous signal into valid timeslots of a frame of said container signal and providing stuff bits into invalid timeslots,
 - wherein said invalid timeslots are uniformly interspersed across said frame.
2. A method as claimed in claim 1, wherein said container signal is a SONET/SDH signal, and said synchronous network is a SONET/SDH network.
3. A method as claimed in claim 2, wherein said SONET/SDH signal further comprises a synchronous tributary.
4. A method as claimed in claim 2, wherein said SONET/SDH signal comprises a plurality of transparent tributaries.
5. A method as claimed in claim 1, wherein said invalid timeslots comprise one of a fixed stuff and an adaptive stuff bit.
6. A method as claimed in claim 5, wherein said step of distributing comprises:
 - receiving a continuous stream of data bits and determining the phase difference between said arbitrary rate R1 and said rate R;
 - adding to said continuous stream a definite number of timeslots for accommodating said fixed stuff bits within said frame, and an adjustable number of timeslots for accommodating said adaptive stuff bits within said frame, based on said phase difference.
7. A method as claimed in claim 6, wherein said adjustable number is substantially larger than said definite number.

8. A method as claimed in claim 6, wherein said definite number includes transport overhead TOH timeslots and reminder fixed stuff bits timeslots.

9. A method as claimed in claim 8, further comprising providing maintenance, operation, administration and provisioning information in said TOH timeslots.

10. A method as claimed in claim 6, wherein said step of adding comprises:

partitioning said frame into a number of equally sized data blocks and said definite number of timeslots;
for each block,

determining the number of fixed stuff bits and evenly distributing said fixed stuff bits within said block;

determining a control function β indicative of said adjustable number; and

evenly mapping said fixed stuff bits and said adaptive stuff bits uniformly within a next block based on said control function.

11. A method as claimed in claim 10, wherein said step of mapping comprises:

providing a counter C for identifying a timeslot in said block;

defining the binary bit reversal α of said control function β ;

calculating the bitwise transition delta of said counter C ; and

determining if a timeslot identified by said counter C is an invalid timeslot, whenever a function $\text{Valid}(C, \beta)$ is false; and

providing an adaptive stuff bit into said invalid timeslot.

12. A method as claimed in claim 1, further comprising recovering said continuous signal from said synchronous signal at a receive site, by extracting the data bits of said continuous signal from said valid timeslots of said frame.

13. A synchronizer for mapping a continuous format signal of an arbitrary rate for transport over a synchronous network as a transparent tributary signal, comprising:

a data recovery unit for receiving said continuous format signal and recovering a stream of data bits and a data clock indicative of said arbitrary rate;

a receiver buffer unit for receiving said stream of data bits, determining a phase difference between said arbitrary rate and the rate of a frame of said tributary, and generating a control function β ;

a mapping unit for extracting said stream of data bits from said receiver buffer unit at a mapping clock rate, and uniformly distributing a count of stuff bits and data bits into said frame at a block clock rate according to said control function β .

14. A synchronizer as claimed in claim 13, wherein said receiver buffer unit comprises:

an elastic store for temporarily storing an amount of data bits of said stream at said data clock and providing said data bits to said mapping unit at said block clock rate;

a digital PLL for determining the phase difference between said arbitrary rate and said mapping clock and providing said control function β .

15. A synchronizer as claimed in claim 13, wherein said data recovery unit comprises a frequency agile PLL for detecting said arbitrary rate, and a receiver for detecting said data bits using said data clock.

16. A synchronizer as claimed in claim 13, wherein said mapping unit comprises:

a block clock gapper for receiving a clock indicative of the rate of said synchronous frame and providing said block clock of a block rate accounting for all timeslots of said synchronous frame and with gaps accounting for a definite number of timeslots for accommodating fixed stuff bits;

a mapping clock gapper for receiving said block clock and said control signal β and providing a mapping clock of a mapping rate accounting for all timeslots of said synchronous frame and with gaps

accounting for an adjustable number of timeslots for accommodating adaptiv stuff bits within said fram ; and

a mapper for receiving said block clock and said mapping clock and accordingly mapping said stream of data bits in said frame.

17. A synchronizer as claimed in claim 13, further comprising a receiver OH FIFO for re-arranging a plurality of transport overhead TOH timeslots for seamless transport of said frame within said synchronous network.

18. A synchronizer as claimed in claim 17, further comprising an overhead multiplexer for adding operation, administration, maintenance and provisioning data into said TOH timeslots.

19. A de-synchronizer for reverse mapping a continuous format signal of an arbitrary rate received over a synchronous network as a transparent tributary signal, comprising:

a reverse mapping unit for receiving a frame of said tributary at a block clock rate and a control function β , and extracting a stream of data bits at a mapping clock rate, while excluding stuff bits according to said control function β ;

a transmitter buffer unit for receiving said data bits, and determining a phase difference between said arbitrary rate and the rate of said frame; and

a data transmit unit for receiving said data bits and transmitting said continuous format signal at a data rate controlled by said phase difference.

20. A de-synchronizer as claimed in claim 19, wherein said control function β is received in said frame.

3. Detailed Description of Invention

Field of the Invention

The invention is directed to transparent transportation of data signals with a continuous format, and in particular to a method for mapping arbitrary continuous signals into a SONET frame.

Background Art

Rapid provisioning of a customer's requested service is a valuable network function. There can be a large range of possible bit-rates for such services, or indeed the service and its bit-rate may not even have been defined when the network equipment is installed. Therefore, rapid provisioning of a service of arbitrary bit-rate is a valuable function.

Data transmission formats can be divided into SONET, other continuous formats, and burst formats. Burst formats do not have a continuous clock, transmission of such signals do not require any given phase relationship between bursts. On the other hand, the phase of the clock of continuous formats has continuity under normal conditions, and the frequency of the clock is bounded. Examples of such bounds are ± 20 ppm (parts per million of the bit rate) and ± 100 ppm.

The dominant signal format in the fiber optic networks follows the synchronous standard SONET in North America and SDH elsewhere. In this specification, SONET is defined to include SDH. SONET enables multiplexing, adding and dropping, and general transportation of signals. For a service, being able to be easily transported by a SONET network is a valuable attribute, in that it enables the network providers to make use of the large base of installed SONET-compatible equipment.

SONET is a physical carrier technology, which can provide a transport service for ATM, SMDS, frame relay, T1, E1, etc. As well, operation, administration, maintenance and provisioning (OAM&P) features of SONET provide the ability to reduce the amount of back-to-

back multiplexing, and more importantly, network providers can reduce the operation cost of the network.

The SONET standards ANSI T1.105 and Bellcore GR-253-CORE, define the physical interface, optical line rates known as optical carrier (OC) signals, a frame format, and an OAM&P protocol. Opto/electrical conversion takes place at the periphery of the SONET network, where the optical signals are converted into a standard electrical format called the synchronous transport signal (STS), which is the equivalent of the optical signal. Namely, the STS signals are carried by a respective optical carrier, which is defined according to the STS that it carries. Thus, an STS-192 signal is carried by an OC-192 optical signal.

The STS-1 frame consists of 90 columns by 9 rows of bytes, the frame length is 125 microseconds. A frame comprises a transport overhead (TOH) occupying 3 columns by 9 rows of bytes, and a synchronous payload envelope (SPE) occupying 87 columns of 9 rows of bytes. The first column of the SPE is occupied by path overhead bytes.

As such, an STS-1 has a bit rate of 51.840 Mb/s. Lower rates are subsets of STS-1 and are known as virtual tributaries (VT), which may transport rates below DS3. Higher rates, STS-N, where $N=1, 3, 12, \dots, 192$ or higher, are built by multiplexing tributaries of a lower rate, using SONET add/drop multiplexers. An STS-N signal is obtained by interleaving N STS-1 signals. For example, an STS-192 is made of 192 STS-1 tributaries, each separately visible, and separately aligned within the envelope. The individual tributaries could carry a different payload, each with a different destination.

The STS-N has a TOH made of all N TOHs of the individual tributaries, and a SPE made of all N SPEs of the tributaries, each with its own POH.

Some services, that operate at a higher rate, are transmitted in an STS-Nc signal (c for concatenation). The STS-1s into the STS-Nc signal are kept together. The whole envelope of the STS-Nc signal is routed, multiplexed and transported as a single entity rather than as N individual entities. The TOH and the start of the SPE for the N constituents are all

aligned, since all the constituents are generated by the same source, with the same clock. The first STS-1 in the concatenated signal carries the single set of POH, all that is required for an STS-Nc.

Mapping of one rate or format into another is well known. Bellcore TR-0253 describes in detail the standard mappings of the common asynchronous transmission formats (DS0, DS1, DS2, DS3, etc) into SONET. Similar mappings are defined for the ETSI hierarchy mapping into SDH. Optical transmission equipment has mapped one proprietary format into another. For example, FD-565 could carry Nortel's FD-135 proprietary format as well as the DS3 standard format.

However, the standards or proprietary schemes allow transportation of a very specific set of signals, with format specific hardware. These methods of mapping cannot be used to map rates that vary significantly from the standard. Furthermore, these mappings are each precisely tuned for a particular format and a particular bit-rate, with e.g. a ± 20 ppm tolerance. If a signal has, for example, a bit rate even 1% different than that of a DS3, cannot be transported within SONET. In addition, a different hardware unit is generally required to perform the mapping of each kind of signal.

A solution to the above problem is to add a "wrapper" to an arbitrary continuous signal. The rate of the resulting signal is a function of the signal being wrapped. Namely, a 1Mb/s wrapper added to a signal of rate X produces a format with rate $X+1$ Mb/s. A variation on this adds a percentage of X. For example, a common line coding 8B/10B produces a format with a rate of 112.5% of X. As such, the "wrapper" methods do not produce formats that have a pre-defined fixed bit rate for arbitrary inputs. The resulting signal cannot in general be time multiplexed to be transported on a high speed network.

United States Patent No. 5,784,594 (Beatty) proposes a "TDM Wrapper" format where an arbitrary signal is mapped into as much of a frame as required, and the rest of the frame is left empty. However, this method requires a very large memory for each direction of conversion to

hold the bits while waiting for the appropriate time slot to transmit them. As a result, this format is expensive to implement with high speed signals.

Packet or cell based formats map arbitrary input streams into SONET and SDH. While adequate for packet systems, these methods do not meet the jitter or wander requirements of most continuous signal formats due to the "one size fits all" mapping methods used. The clock phase information of the input signal is effectively eliminated in these methods, and so cannot be transmitted.

US Patent Application 09/307812 (Solheim et al., entitled "Protocol Independent sub-rate device" filed on May 10, 1999 and assigned to Nortel Networks Corporation) discloses a method of transporting different type of clients (IP, ATM, SONET, Ethernet, etc.) together. The '812 application discloses time-multiplexing lower speed (subrate) channels of arbitrary rates and formats into a single higher speed channel, and then demultiplexing the channels at the far end of the system. The portion of the bandwidth assigned to any given subrate channel can be provisioned without any change to the hardware or software. This significantly simplifies and speeds the provisioning of these services by the carrier. Tributaries with new protocols can be accommodated as well, significantly speeding up the delivery of support for these new protocols.

There remains a need for an efficient method and apparatus that will map arbitrary signals into SONET such that the signals can be recovered with low timing jitter at low cost.

SUMMARY OF THE INVENTION

It is an object of the present invention to map arbitrary signals having a continuous format into a SONET frame. This allows any qualified format to be transparently transported within a SONET network.

Accordingly, the invention comprises a method for transmitting a continuous digital signal of an arbitrary rate R_1 over a synchronous network as a transparent tributary, selecting a fixed length container signal of a rate R , where R is higher than the arbitrary rate R_1 of the continuous signal, and at a transmit site, distributing the bits of the continuous signal

into valid timeslots of a frame of the container signal and providing stuff bits into invalid timeslots, wherein the invalid timeslots are uniformly interspersed across the frame.

The invention further comprises a synchronizer for mapping a continuous format signal of an arbitrary rate for transport over a synchronous network as a transparent tributary signal, comprising a data recovery unit for receiving the continuous format signal and recovering a stream of data bits and a data clock indicative of the arbitrary rate, a receiver buffer unit for receiving the stream of data bits, determining a phase difference between the arbitrary rate and the rate of a frame of the tributary, and generating a control function β , a mapping unit for extracting the stream of data bits from the receiver buffer unit at a mapping clock rate, and uniformly distributing a count of stuff bits and data bits into the frame at a block clock rate according to the control function β .

According to another aspect of the invention there is provided a desynchronizer for reverse mapping a continuous format signal of an arbitrary rate received over a synchronous network as a transparent tributary signal, comprising, a reverse mapping unit for receiving a frame of the tributary at a block clock rate and a control function β , and extracting a stream of data bits at a mapping clock rate, while excluding stuff bits according to the control function β , a transmitter buffer unit for receiving the data bits, and determining a phase difference between the arbitrary rate and the rate of the frame, and a data transmit unit for receiving the data bits and transmitting the continuous format signal at a data rate controlled by the phase difference.

Advantageously, the method of mapping according to the invention allows use of a common technology, such as SONET, for transparently transporting tributaries of same or different formats. Almost any continuous format could be transported by SONET using this novel mapping, without changing any bit. Another advantage of the present invention is that the jitter or wander added by the method is minimal.

The synchronized/desynchronizer according to the invention handles signals whose format is unknown at the time of design, as long as the jitter tolerance and generation specifications are compatible with the very accommodating range designed into the unit. This is a proprietary mapping, that is designed on the fly by the trib software, and is communicated within the channel to the corresponding trib at the far end.

DESCRIPTION OF THE PREFERRED EMBODIMENT

The mapping system according to the invention maps a digital signal that has a constant line rate, up to the designated maximum capacity, into a SONET envelope of a provisioned size. A mapping function could be performed in a tributary unit of a SONET transport shelf, and the reverse mapping function (also called de-mapping) could be performed in a similar unit at the far end of a SONET connection.

Figure 1A shows a block diagram of an exemplary transmission system with the mapping system according to the invention, transparently transporting a plurality of services over a SONET network. Only one

direction of transmission, as shown by the arrows, is illustrated on this figure for simplification.

Signals $S_1, \dots, S_j, \dots, S_n$, where n is the number of the tributaries and j is the range of a tributary, are carried between two sites **A** and **B** into a SONET signal S , over a SONET network 7. Signals S_1-S_n are digital signals of a continuous format, and are treated at the nodes **A** and **B** as tributaries of SONET signal S . We also note the rate of each signal S_j with R_j and the rate of signals S with R . Signals S_j can carry the same or different type of services. Each trib receiver $1-1_n$ recovers the data bits for the respective continuous format signal S_1-S_n . Node **A** is provided with one or more synchronizers 20_1-20_n , each synchronizer 20_j for mapping the data bits of corresponding trib signal S_j into a SONET envelope of an appropriate size.

The size of the frame to carry transparently the continuous signals between certain transmission nodes is selected in software, and is provisioned having in view a satisfactory bandwidth usage for a large range of continuous format signals. For example, if a $nxSTS-12$ is used for the envelope, n is between four and twenty for a high speed variant of the synchronizer, and between one and five for a medium speed variant. This avoids stranding significant network capacity.

After each signal was mapped into a respective SONET envelope, the tributaries are multiplexed by a SONET multiplexer 3 into a high rate signal S , which is launched by a SONET transmitter 5 over optical network 7 towards site **B**.

The reverse operation is performed at site **B**. Namely, the optical receiver 9 recovers the data in signal S , demultiplexer 3' separates the signals and presents them to a respective de-synchronizer $40-40_n$. Each de-synchronizer 40_j re-arranges the bits in the respective format associated with the signal S_j , which is presented to a trib transmitter 11_j . Each trib transmitter $11-11_n$ launches the respective signal S_1-S_n on the associated trib network, or to an associated end user.

An example of a mapping algorithm is provided next for an STS-192c signal, for showing the basic concepts and the feasibility. Other envelopes can also be used, the invention not being limited to the STS-192c signals.

Figure 1B shows an STS-192c frame 1, comprising TOH 2, and the STS-192c SPE (synchronous payload envelope) 4. The payload comprises $192 \times 87 \times 9 \times 8 = 1,202,688$ bits.

A block 10-J is defined herein as a 1056-bit field, which comprises data bits, fixed stuff bits and adaptive stuff bits, as it will be seen later. An STS-192 SPE can accommodate 1138 such blocks 10-1 to 10-K (where $K=1138$), that occupy the area shown in grey and designated by reference numeral 8. Block field 8 has 1,201,728 bits. The remaining 960 bits in the envelope 4 are comprised by the POH bits 6 ($9 \times 8=72$ bits) and a remainder field 14 of 888 bits. The number of the bits in fields 6 and 14, is unchanged, irrespective of the rate $R1$ of the continuous format signal mapped into the SONET frame 1. Therefore, these bits are called in the following fixed stuff bits.

On the other hand, the number of stuff bits necessary to fill the block field 8 varies function of the rate $R1$ of the continuous format signal $S1$. These stuff bits are called herein adaptive stuff bits.

According to the invention, the data bits of the signal $S1$ are mapped into frame 1 with evenly interspersed fixed stuff bits and adaptive stuff bits. These stuff bits are distributed uniformly within each block, on the fly, since the rate $R1$ may not be known in advance. Therefore, the synchronizer defines a valid location, that is a location for a data bit, and an invalid location, that is a location for a stuff bit for the next block, based on phase information accumulated when the data bits of the current block are mapped. In addition, the synchronizer also distributes evenly the overhead bits at the time of the actual mapping, but realigns these in the timeslots provided according to the SONET standard after mapping operation, so that the frame is recognized by the SONET equipment. At the far end, the synchronizer effects the reverse operation, by absorbing

th fixed stuff bits and the adaptive stuff bits, so that the data bits can be reverse-mapped to regen rate $S1$.

It is to be noted that Figure 1B shows the structure of a frame intuitively; in accordance with this invention the mapping algorithm distributes the fixed stuff bits and the adaptive stuff bits uniformly within the frame 1. We also note that the above calculations are applicable to a STS-192c frame; similar consideration apply to other SONET signals.

The bits in each block are allocated as shown in Figure 1C. A block 10-1 comprises a data field 17 having 1023 ($2^{10}-1$) bits for data, a control field 13 having 16 bits, and a spare field 15 having 17 bits for future use.

The 1023 bits of field 17 provide a bit rate of 9,313.392 Mbps ($1023 \times 1138 \times 8000$) for transportation of the data into a STS-192c frame. The size of the frame is provisioned and fixed for a certain application, i.e. tributaries of arbitrary rates are mapped in frames of a same size. The mapping technique is adaptive for any trib, rather than a different frame being used for every trib. If a trib has a lower rate than 9,313,392 Mbps, it must be justified into the STS-192c, changing more of the bits of data field 8 into stuff bits. Figure 1C illustrates field 19 within field 17, of a variable size v , which size is determined during mapping by comparing the phase between the clock of signals $S1$ to that of signal S .

Field 13 comprises a 10-bit control function β . The size of β was selected in accordance with the size of the block, so as to uniquely determine the position of valid bits in the next block, according to the adaptive stuffing algorithm below. A 10-bit number can assume 1024 values, which is one more than the size of a block. The value of β also gives the number of the valid bits in the next block. The additional 6 bits of field 13 are necessary for single bit error correction and multiple error detection.

In case of detection of multiple errors, the β from the previous block is used as the default, for fast reframing downstream with a minimal PLL transient. The bits of fields 15 and 19 are interspersed within the block.

The value of β may change between adjacent blocks, as not all blocks have the same number of adaptive stuff bits, but β remains constant within each block.

The adaptive stuffing algorithm defines the binary bit reversal of β which is denoted with α . That is, the most significant bit of β becomes the least significant bit of α ; similarly the least significant bit of α becomes the most significant bit of β . Table 1 details this translation by way of examples.

Table 1. Determination of α

β										α									
β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}	β_{10}	β_9	β_8	β_7	β_6	β_5	β_4	β_3	β_2	β_1
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
1	0	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0	1
0	1	1	1	0	0	1	0	0	0	0	0	0	1	0	0	1	1	1	0

α is also determined on a per block basis, and as in the case of β , the value of α may change between the adjacent blocks but does not change within a block.

Also defined herein is a counter C , and a value D . C is the counter of bits in a block, and is represented by a 10 bits binary number. C increments from 1 to 1023, and as such identifies the timeslot occupied by a bit in the block.

D is the bit-wise transition delta of C , and is represented by a 10 bit binary number with exactly one bit set. This set bit is in the position of a 0 - to - 1 transition that occurs when counter C advances with one bit. Using Boolean functions, each bit of D is given by the bits of range n and $n-1$ of counter C , according to the equation:

$$D_n = C_n \text{ AND NOT } (C-1)_n$$

EQ1

Table 2 gives examples of the values assumed by D for a given value of C .

Table 2. Determination of D for a given C

C_{12} , Counter of bits in the block											D_{12} , Bit-wise transition Δ of C									
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}
$C=1$	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
C	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
$C=1$	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0
$C=2$	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1

In order to spread stuff bits more or less evenly among valid data in the block, the adaptive stuffing algorithm according to the invention states that a bit is valid when a bit in D is also set in α .

This can be described as in Eq2, for the C^n bit of a block:

Valid(C, β), if any bit of ($\alpha_{1,2,\dots} \text{ AND } D_{1,2,\dots}$) is non zero Eq2

A valid bit corresponds to a bit which is assigned to data, and consequently an invalid bit corresponds to a stuff bit. Table 3 shows a simple example of how the algorithm works for a block with 7 bits, for which size of β is 3 bits. It is to be understood that the algorithm operates in a similar way for blocks of 1023 bits and a 10-bit β , but it is impractical to detail the full stuffing sequences in this specification.

The entries in Table 3 are the result of the binary function Valid(C, β). The columns illustrate how data and stuff bits are interspersed for a particular number of valid bits in the block, as given by β .

For each value of C where Valid(C) is true, a valid data bit is present in the timeslot identified by C , for each untrue value of Valid(C), a stuff bit is placed into the timeslot. Using this scheme, the invalid stuffing bits are spread almost uniformly through the frame.

Table 3. Example of flexible mapping for a 7-bit block, for a 3-bit β

	β		000	001	010	011	100	101	110	111
	α		000	100	010	110	001	101	011	111
C	D		Valid(C,0)	Valid(C,1)	Valid(C,2)	Valid(C,3)	Valid(C,4)	Valid(C,5)	Valid(C,6)	Valid(C,7)
1	001	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
2	010	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
4	100	100	Stuff	Data	Stuff	Data	Stuff	Data	Stuff	Data
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
6	110	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data

In this example, counter C counts from 1 to 7 and D is evaluated in the respective column, for all values of C . Each value of D , as C increments, is then compared with α . If the set bit in D is also set in α , the corresponding C^{th} bit in the block will be a data bit. If the set bit in D is not set in α , the corresponding C^{th} bit in the block will be a stuff bit.

Let's take as an example a block where the bit rate $R1/R$ is $5/7^{\text{th}}$ of the available capacity, which means that β is binary 5 (101), and α , the binary bit reversal of β , is also 5 (101). The sequence of data and stuff bits in the block is as per column Valid(C,5) and is:

Data, Stuff, Data, Data, Data, Stuff, Data

It is also apparent on Table 3 that for Valid (C,5) β , which is five, is also the number of valid bits, and the invalid bits are spread almost uniformly through the frame.

Table 4 details the stuffing sequence for 5 consecutive blocks, with a slightly different β between blocks. In this Table, counter C counts from 1 to 7, and two consecutive blocks are shown with a different background (grey and white) for clarity.

Tabl 4. Adaptive Stuff algorithm for five consecutive 7-bit blocks,

		β	000	001	010	011	100	101	110	111
		α	000	100	010	110	001	101	011	111
C	D		Valid(C,0)	Valid(C,1)	Valid(C,2)	Valid(C,3)	Valid(C,4)	Valid(C,5)	Valid(C,6)	Valid(C,7)
1	001	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
2	010	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
4	100	100	Stuff	Data	Stuff	Data	Stuff	Data	Stuff	Data
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
6	110	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
1	001	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
2	010	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
4	100	100	Stuff	Data	Stuff	Data	Stuff	Data	Stuff	Data
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
6	110	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
1	001	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
2	010	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
4	100	100	Stuff	Data	Stuff	Data	Stuff	Data	Stuff	Data
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
6	110	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
1	001	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
2	010	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
4	100	100	Stuff	Data	Stuff	Data	Stuff	Data	Stuff	Data
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
6	110	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
1	001	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
2	010	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data

3	011	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
4	100	100	Stuff	Data	Stuff	Data	Stuff	Data	Stuff	Data
5	101	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data
6	110	010	Stuff	Stuff	Data	Data	Stuff	Stuff	Data	Data
7	111	001	Stuff	Stuff	Stuff	Stuff	Data	Data	Data	Data

As indicated above, β is set for each block, but may vary between subsequent blocks, since the ratio between the total number of stuff bits and the number of blocks may not be an integer. Let's consider the following example:

block 1 (grey) has a β of 5

block 2 (black) has a β of 6

block 3 (grey) has a β of 5

block 4 (black) has a β of 5

block 5 (grey) has a β of 6

In this case, the spreading of stuff bits among data bits is as follows:

DSDDSD DDDSD DSDSD DSDSD DDDSD

where D stands for data, and S for Stuff. To avoid confusion between D and S above, they are written in regular characters, while italic characters are used for signals *S*, and for *D*, bit-wise transition Δ of *C*.

It is again evident from Table 4 that the stuff bits are spread more or less evenly among the data bits, even with a slight variation of β .

Figure 2 illustrates a block diagram of a trib synchronizer 20.

Transparency is obtained as discussed above, by filling a SONET SPE with data received at an arbitrary rate. The data path is illustrated using wide arrows, and reference numerals 22 and 22'. Signal *S_r*, of a continuous format and rate *R₁* is detected by a data recovery unit 36. The data bits then pass through a fill control unit 38, a mapping unit 30, a receiver overhead FIFO (first-in, first-out) 31, and an overhead multiplexer 33. The signal output by synchronizer 20 is now in a SONET frame. It is to be understood that signal *S* has a SONET-type overhead (TOH and

POH) with the respective OAM&P information, and has a SONET rate R, while the placement of the bits into the synchronous payload is according to the mapping algorithm rather than to the SONET standard.

Synchronizer 20 manipulates four different clocks: a data clock 24, a block clock 26, a mapping clock 32 and SONET clocks 28 and 28A. Clock 28 has the STS-192 rate and clock 28A has the rate of the frame. The data clock 24 (rate R1) is recovered from the incoming data by data recovery unit 36 which comprises a receiver 21 and a flexible clock recovery circuit 25. Flexible clock recovery circuit 25 is capable of clock recovery over a broad continuous range of bit-rates. An example of such a circuit is disclosed in the co-pending US Patent application 09/218053 filed on December 22, 1998, entitled "Apparatus and Method for Versatile Digital Communication, by Habel et al., assigned to Northern Telecom Limited. Patent application '053 is incorporated herein by reference.

A certain set of known signal formats could be recognized by an off-line framer 39, shown in dotted lines on Figure 2, and frame and BER performance information reported. Also, line coding could be removed from some signals at the receiver and added on at the transmitter, for better efficiency in mapping. These options depend on the particular type of service, and therefore are not discussed in further detail here.

A receiver buffer unit 38 comprises an elastic store 23 and a receiver digital PLL 29. The data clock 24 is used to clock the input of data into elastic store 23, which is emptied under control of mapping clock 32. Mapping clock 32 is a gapped clock, derived from the STS-192 clock 28. This clock is discontinued at appropriate phase instants as determined by the mapping algorithm, in addition to the gaps in the block clock. In this way, data 22 is synchronized to the mapping frequency at the input of mapper 27.

If the payload field 4 were filled continuously with data bits to the capacity required, and the remaining capacity were a continuum of stuff bits, the elastic store 23 fill would vary widely, requiring a rather large depth for the store 23. The elastic store 23 would fill rapidly while stuff bits are loaded, emptying rapidly while a continuous stream of tributary data

bits are loaded. This situation is avoided in the configuration of Figure 2, where the elastic store 23 is emptied at substantially regular intervals by mapping clock 32.

On the other hand, the elastic store 23 must be sufficiently deep to absorb all input jitter and wander from the trib. If the fill of the elastic store 23 is sufficiently well controlled, it can be guaranteed never to overflow or underflow even in the presence of worst-case jitter and wander, and the synchronizer 20 will still meet jitter tolerance requirements.

Experimentally, the minimum size of the elastic store 23 was determined at 256 bits.

Receiver digital PLL 29 controls the rate at which the elastic store is emptied to maintain the optimal fill by way of β which determines the mapping clock 32. In other words, the average rate of mapping clock 32 is controlled to track the average rate of data clock 24 and β results from the phase difference between these clocks. As indicated above, β controls filling of the next block. This control has the advantage that the synchronizer 20 does not need a pointer adjustment as per SONET standard. Rather, control of the elastic store fill ameliorates any line and trib rate variations with time (line and trib jitter and wander), as long as the maximum trib rate never exceeds the payload rate.

To determine β , the input to the elastic store 23 is sampled periodically and phase information on data 22 is input to PLL 29. The digital PLL 29 may for example comprise a 24-bit accumulator. At the start of the block, the fill of the elastic store 23 given by counter C of mapper 27 is latched relative to e.g. 50%. Then, the phase of sample 34 is incremented into the accumulator and added to the phase left-shifted by 3 bits. The upper 10 bits of this sum S is β . The accumulator must be clipped at FFFFFFFF to not roll-over, and clipped at a lowest value such as 400000 to reflect the low frequency limit of the analog output PLL range. Other implementations of digital PLL are also possible.

If the elastic store 23 starts to overfill, β would be increased to empty the store by speeding-up the mapping clock 32. Similarly, if store

23 begins to empty, β would be decreased to allow store 23 to fill. The target fill is preferably 50%.

Mapping unit 30 comprises a block clock gapper 37, a mapping clock gapper 35 and a mapper 27.

Block clock gapper 37 receives the STS-192 clock 28 which features gaps and regular cycles accounting for SONET TOH. Clock 28 generates the block clock 26, which has in the above example 1138 (number of blocks) * 1056 (size of a block) = 1,201,728 cycles per SONET frame, with 42,432 gaps spread evenly through the frame. As indicated above, the gaps of the block clock 26 are due to the SONET overhead, namely field 2 in Figure 1B, whose size is $3 * 9 * 8 * 192$, and to the fixed stuffing, namely fields 6 and 14, whose size is 960. Block clock 26 represents the total allocation of bits in field 8. In other words, block clock 26 is discontinued at approximately every 30^{th} bit to keep room for the TOH bits, POH bits and the fixed stuff bits, in the case when the overhead size of the frame is according to the SONET standard.

Mapping clock gapper block 35 has the same rate as the block clock, but is further gapped under control of β , as described above, with a pulse at every valid bit location, to further account for the adaptive stuff bits resulted from the difference between rates R1 and R.

Mapper 27 takes mapping clock 32, block clock 26 and some other complementary clocks, not shown for simplicity, and justifies data 22 using both fixed and adaptive stuff bits. The data bits are clocked out from elastic store 23 into the mapper using mapping clock 32. The data bits, fixed and adaptive stuff bits are clocked out from mapper 27 using block clock 26. The mapper 27 has essentially no memory, the elastic store 23 and the FIFO 31 representing all the memory of the synchronizer.

The bits from mapper 27 denoted with reference numeral 22', as they comprise data, fixed and adaptive stuff bits, are clocked into the receiver overhead FIFO (first-in first-out) 31, which reserves timeslots for for the SONET overhead locations. Next, bits 22' are clocked out of FIFO 31 with clock 28A, whereby FIFO 31 is reset synchronously once each

frame. The depth of FIFO 31 has only to be sufficient to store payload bits during the phase instants of the frame when frame OH is being clocked into the OH MUX. If the frame has the same OH to payload ratio as SONET, this depth must be greater than $192 \times 8 \times 9 \times 3$ bits, and is preferably larger than $192 \times 8 \times 12 \times 3$ bits.

From FIFO block 31, bits 22' are clocked into the SONET overhead multiplexer 33 where the SONET overhead is added in the respective empty timeslots and the signal is then treated as an STS-192. The SONET clocks 28 and 28A, shown in thin lines, are locked to the rest of the shelf, in the usual manner.

A serial hardware implementation is described for simplicity. Parallel implementations of this kind of mapping, such as byte wide implementations can obtain lower clock speeds. These parallel implementations can have staggered block alignments for reduced jitter. DSP control rather than hardware control would give greater freedom to optimize the PLLs.

Figure 3 shows the block diagram of the transmitter side of the transparent reverse synchronizer, or desynchronizer 40. The desynchronizer 40 performs the inverse function performed by the synchronizer 20, in a very similar manner, and is provided with similar blocks.

A SONET overhead demultiplexer 53 delineates the SONET overhead from signal 42' using STS-192 clock 28, which is locked to the rest of the shelf in the usual manner. A transmitter overhead FIFO 51 is reset synchronously once each frame with clock 28A. Overhead FIFO 51 absorbs overhead locations so as to present to the mapper 47 the data 42' received in the payload, comprising data bits together with the fixed stuff bits and adaptive stuff bits.

If pointer adjustment is not considered, the transmitter OH FIFO 51 could have a similar depth with that of receiver OH FIFO 31 of synchronizer 20. For example, if the SONET OH is used for the frame, the required depth is, as in the case of FIFO 31, $192 \times 8 \times 12 \times 3$ bits. FIFO

could thus store sufficient data bits so that transmitter OH FIFO 51 is not emptied during the frame phase when the OH is being demultiplexed from the bit stream. However, as pointer alignment is necessary for the desynchronizer, the transmitter OH FIFO 51 must have additional depth to tolerate the worst-case series of positive or negative pointer adjustment events.

A reverse mapping unit 50 comprises a reverse mapper 47, a mapping clock gapper 55, and a block clock gapper 57.

The block clock gapper 57 gaps the STS-192 clock 28 to create block clock 26. Block clock 26, as in the case of the synchronizer 20, has 1,201,728 cycles per frame, with 42,432 gaps spread evenly across the frame. The gaps account for fields 2, 6 and 14 of Figure 1B. In other words, this clock rejects the TOH and the fixed stuff bits.

Block clock gapper 57 of the desynchronizer also includes/deletes gaps for pointer adjustments. These inclusions must be spread out for three frames to minimize the phase hit from a pointer adjustment.

Mapping clock gapper 55 receives the block clock 26 and β , read from an in-band OH channel within the block. Using β , gapped clock 26 is further gapped to produce the mapping clock 32. Mapping clock 32 gaps-out data bits 42' so that strictly trib data bits 42 are clocked out.

Bits 42 are next processed by a transmitter buffer unit 54, which comprises an elastic store 43 and a transmitter digital PLL 49. The trib data bits 42 are clocked into the output elastic store 43 using mapping clock 32. The elastic store 43 is emptied by the data clock 24, output by the flexible clock recovery circuit 45.

The phase of the output elastic store 43 is sampled periodically by the transmitter digital PLL. The sample 34 is processed digitally, and an output signal passed to the flexible clock circuit 45, to control the voltage of the VCO. The flexible clock circuit 45 is of a similar type with the flexible clock circuit 25 of the synchronizer, and provides data clock 24.

The bandwidth of the Tx PLL 49 must be as low as possible to filter out jitter from the mapping and from pointer adjustments, and yet high enough to suppress the VCO noise.

While the invention has been described with reference to particular example embodiments, further modifications and improvements which will occur to those skilled in the art, may be made within the purview of the appended claims, without departing from the scope of the invention in its broader aspect.

4. Brief Description of Drawings

The foregoing and other objects, features and advantages of the invention will be apparent from the following more particular description of the preferred embodiments, as illustrated in the appended drawings, where:

Figure 1A is a block diagram of a communication network with the mapping system according to the invention;

Figure 1B is a OC-192c frame, showing the blocks according to an embodiment of the invention;

Figure 1C shows an example of the structure of a block;

Figure 2 illustrates a block diagram for a synchronizer according to an embodiment of the invention; and

Figure 3 shows a block diagram for a de-synchronizer according to an embodiment of the invention.

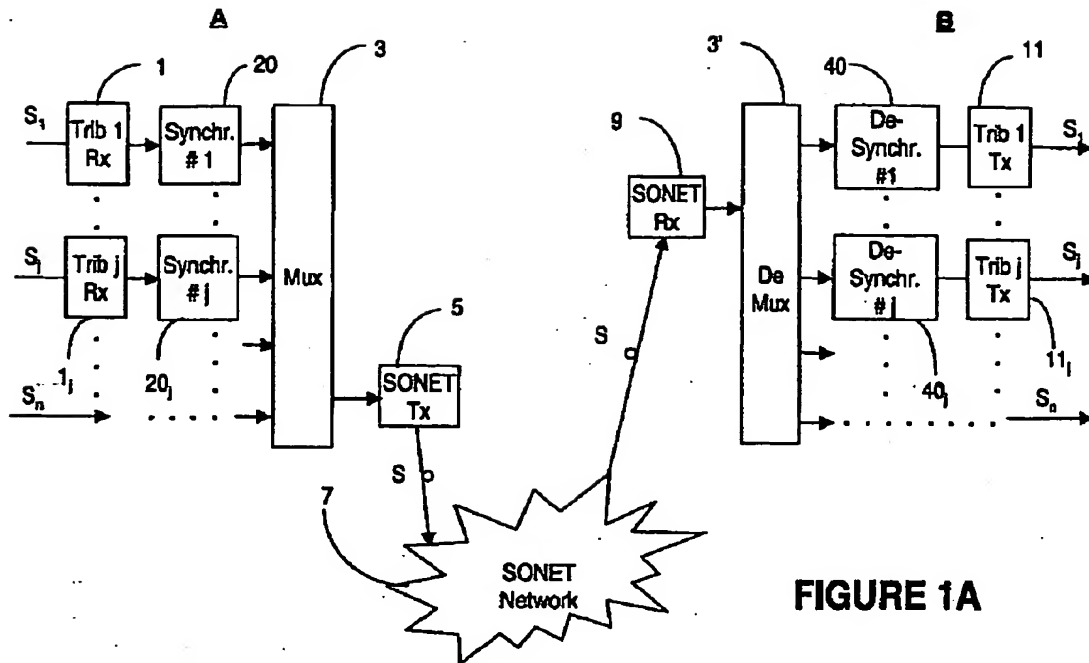


FIGURE 1A

FIGURE 1B

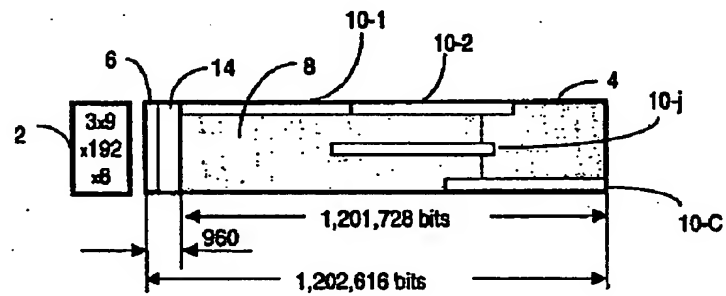


FIGURE 1C

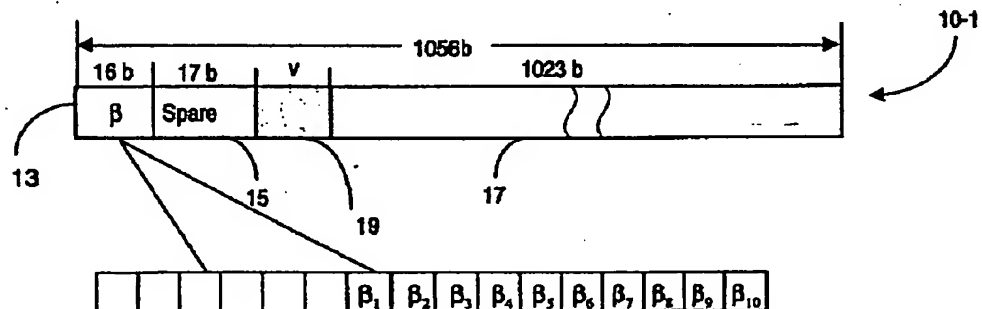
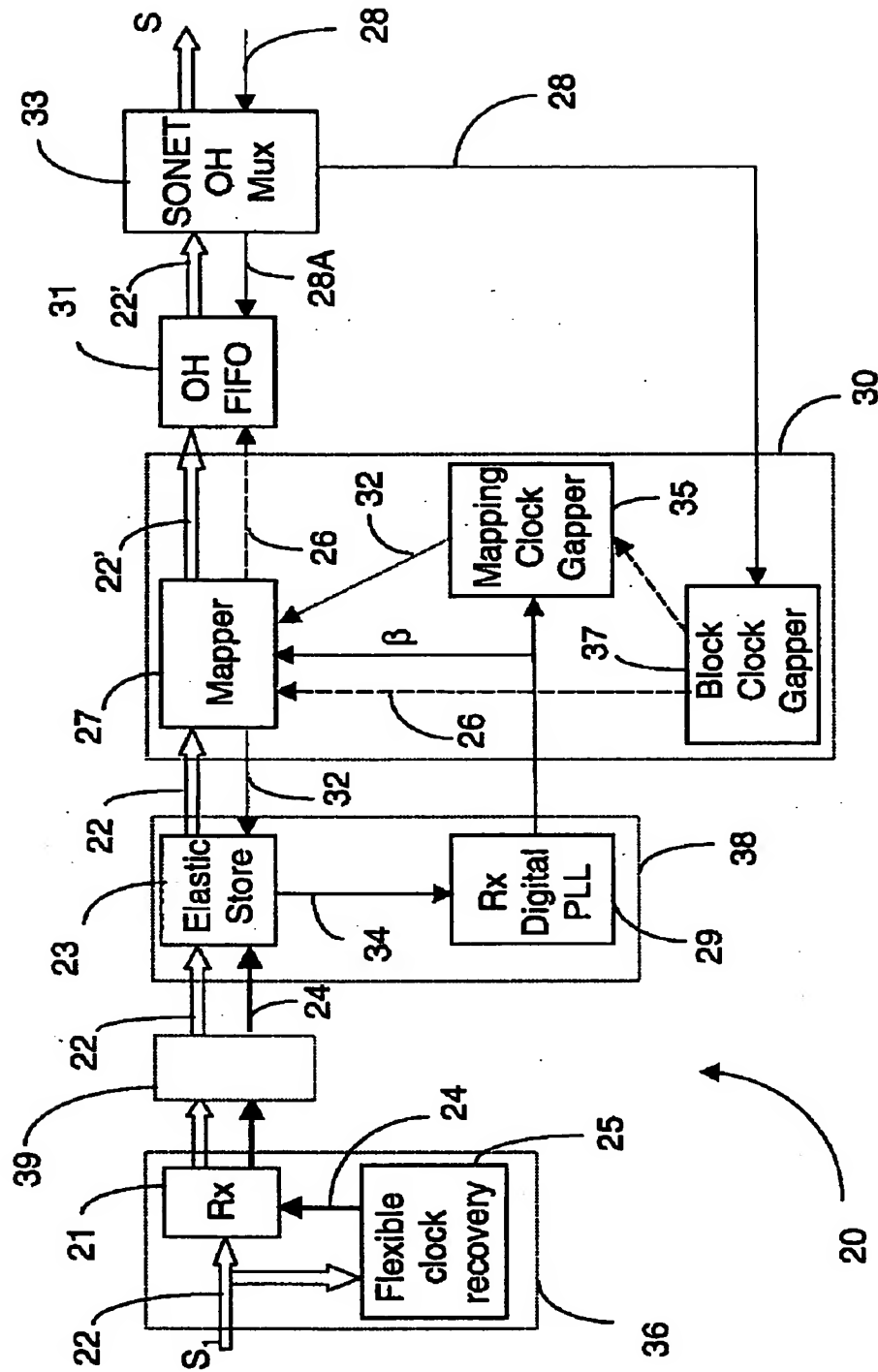


FIGURE 2



A synchronizer/de-synchronizer maps continuous format signals of an arbitrary rate into frames of pre-selected single common rate, such as SONET frames, with no bits changed and very little jitter or wander added. In this way, the continuous format signal may be carried transparently as a tributary of a SONET network. Each frame comprises a definite number of fixed stuff bits, including transport overhead bits and remainder fixed stuff bits. A frame also comprises an adjustable number of adaptive stuff bits, resulting from the phase difference between the arbitrary rate and the common rate. A mapping function is performed in a tributary unit shelf of a SONET transport shelf, and the reverse mapping function is performed in a similar way at the far end of a SONET connection. The stuff bits are spread uniformly within the frame.

2. Representative drawing

Fig. 1A